

# 1 Setting

## 1.1 vimrc

```
1 set ts=4 sts=4 sw=4
2 set ai si nu
```

# 2 Math

## 2.1 basic arithmetic

```
1 typedef long long ll;
2 typedef unsigned long long ull;
3
4 // calculate ceil(a/b)
5 // |a|, |b| <= (2^63)-1 (does not cover -2^63)
6 ll ceildiv(ll a, ll b) {
7     if (b < 0) return ceildiv(-a, -b);
8     if (a < 0) return (-a) / b;
9     return ((ull)a + (ull)b - 1ull) / b;
10 }
11
12 // calculate floor(a/b)
13 // |a|, |b| <= (2^63)-1 (does not cover -2^63)
14 ll floordiv(ll a, ll b) {
15     if (b < 0) return floordiv(-a, -b);
16     if (a >= 0) return a / b;
17     return -(ll)((ull)(-a) + b - 1) / b;
18 }
19
20 // calculate n^k % m
21 ll modpow(ll n, ll k, ll m) {
22     ll ret = 1;
23     n %= m;
24     while (k) {
25         if (k & 1) ret = ret * n % m;
26         n = n * n % m;
27         k /= 2;
28     }
29     return ret;
30 }
31
32 // range modular inverse
33 int modinv[SIZE];
34 void calc_range_modinv(int n, int mod) {
35     modinv[1] = 1;
36     for (int i = 2; i <= n; ++i)
37         modinv[i] = (ll)(mod - mod/i) * modinv[mod%i] % mod;
38 }
```

## 2.2 euler totient function

## 2.3 chinese remainder theorem

# 3 Data Structure

## 3.1 fenwick tree